Hough Transform Neural Network for Seismic Pattern Detection

Kou-Yuan Huang, Jiun-De You, Kai-Ju Chen,
Hung-Lin Lai, and An-Jin Don

Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan
kyhuang2@cs.nctu.edu.tw

Abstract. Hough transform neural network is adopted to detect line pattern of direct wave and hyperbola pattern of reflection wave in a seismogram. The distance calculation from point to hyperbola is calculated from the time difference. This calculation makes the parameter learning feasible. The neural network can calculate the total error for distance from point to patterns. The parameter learning rule is derived by gradient descent method to minimize the total error. Experimental results show that line and hyperbola can be detected in both simulated and real seismic data. The network can get a fast convergence. The detection results can improve the seismic interpretation.

1 Introduction

Hough transform (HT) was used to detect parameterized shapes by mapping original image data into the parameter space [1]-[3]. The purpose of Hough transform is to find the peak value (maximum) in the parameter space. The coordinates of the peak value in parameter space is corresponding to a shape in the image space.

Seismic pattern recognition plays an important role in oil exploration. In a one-shot seismogram in Fig. 1, the travel-time curve of direct wave pattern is a straight line and the reflection wave pattern is a hyperbola. In 1985, Huang et al. had applied the Hough transform to detect direct wave (line pattern) and reflection wave (hyperbola pattern) in a one-shot seismogram [4]. However, it was not easy to determine the peak in the parameter space. Also, efficiency and memory consumption are serious drawbacks.

Neural network had been developed to solve the HT problem [5]-[7]. The Hough transform neural network (HTNN) was designed for detecting lines, circles, and ellipses [5]-[7]. But there is no application to hyperbola detection. Here, the HTNN is adopted to detect the line pattern of direct wave, and hyperbola pattern of reflection wave in a one-shot seismogram. The determination of parameters is by neural network, not by the mapping to the parameter space.

2 Proposed System and Preprocessing

Fig. 1 shows the simulated one-shot seismogram with 64 traces and every trace has 512 points. The sampling rate is 0.004 seconds. The size of the input data is 64x512. The proposed detection system is shown in Fig. 2.

I. King et al. (Eds.): ICONIP 2006, Part II, LNCS 4233, pp. 60 – 69, 2006.

The input seismogram in Fig. 1 passes through the thresholding. For seismic data, $S(x_i, t_i)$, $1 \le x_i \le 64$, $1 \le t_i \le 512$, we set a threshold T. For $|S(x_i, t_i)| \ge T$, data become $\mathbf{x_i} = [x_i\ t_i]^T$, i=1, 2, …, n. Fig. 3 is the thresholding result, where n is 252. Data are preprocessed at first, then fed into the network.

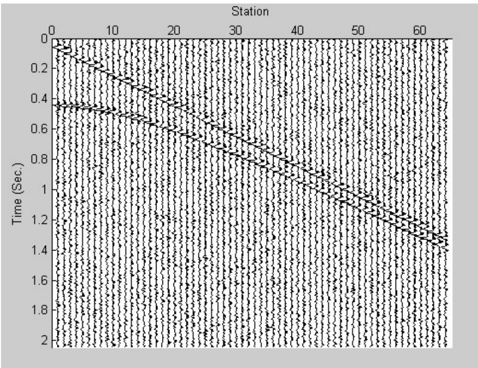

Fig. 1. One-shot seismogram

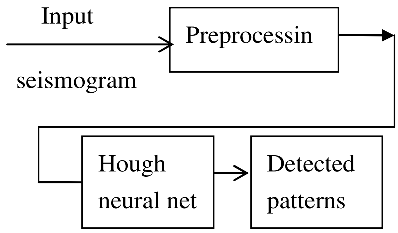

Fig. 2. Detection system for seismic patterns

3 Hough Transform Neural Network

The adopted HTNN consists of three layers: distance layer, activation function layer, and the total error layer. The network is shown in Fig. 4. It is an unsupervised network capable of detecting m parameterized objects: lines and hyperbolas, simultaneously.

Input vector $\mathbf{x_i} = [x_i\ t_i]^T$ is the ith point of the image, where i=1, 2, …, n. In the preprocessed seismic image, x_i is the trace index between shot point and receiving station, and t_i is index in time coordinate. Input each point $\mathbf{x_i}$ into distance layer, we calculate the distance $d_{ik} = D_k(\mathbf{x_i}) = D_k(x_i, t_i)$ from $\mathbf{x_i}$ to the kth object (line or hyperbola), k=1, 2, …, m. Then, d_{ik} passes through the activation function layer and the output is $Er_{ik} = 1 - f(d_{ik})$, where $f(\cdot)$ is a Gaussian basis function, i.e., $f(d_{ik}) = \exp(-\frac{d_{ik}^2}{\sigma^2})$ and Er_{ik} is the error or the modified distance of the ith point related to the kth object. Thus, when d_{ik} is near zero, Er_{ik} is also near zero. Finally, we send Er_{ik} (k=1, 2, …, m) to the total error layer and $Ec_i = C(\mathbf{Er}_i) = C(Er_{i1}, ..., Er_{ik}, ..., Er_{im}) = \prod_{1 \le j \le m} Er_{ij}$ is the total error of $\mathbf{x_i}$ in the network. When $\mathbf{x_i}$ is belonged to one object, then Er_{ik}=0, and Ec_i=0.

We derive the distance calculations from to point to the line detection and the hyperbola as follows.

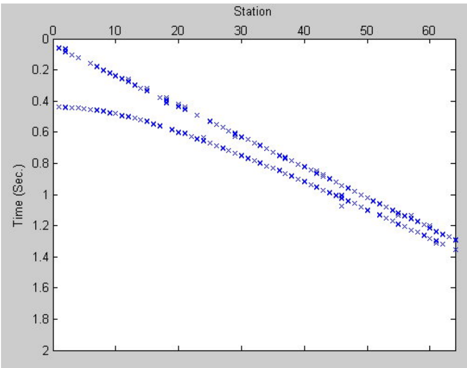

Fig. 3. Result of thresholding

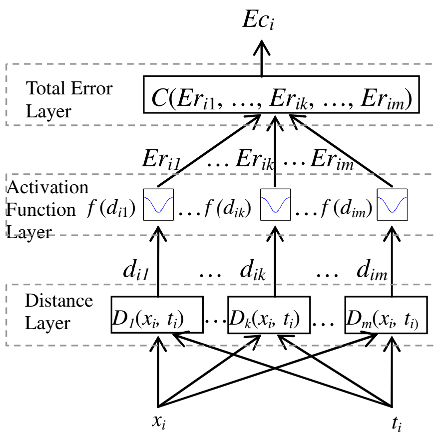

Fig. 4. Hough transform neural network

3.1 Distance Layer

Distance from Point to Line. Although Basak and Das proposed Hough transform neural network to detect lines, they used the second-order equation of conoidal shapes [6]. Here, we use the direct line equation in the analysis.

For line equation $\mathrm{L_k}(\mathbf{x}) = \mathbf{w}_k^T\mathbf{x} + b_k = w_{k,1}x + w_{k,2}t + b_k = 0$ where $\mathbf{w}_k = [w_{k,1} \quad w_{k,2}]^T$, and k is the kth line. We want to find the minimum distance from $\mathbf{x_i}$ to $\mathrm{L_k}(\mathbf{x})$. That is, minimize $\mathrm{Dist}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}_i\|^2$ subject to $\mathrm{L_k}(\mathbf{x}) = 0$.

From Lagrange method, the Lagrange function is $\mathrm{L}(\mathbf{x}, \lambda) = \mathrm{Dist}(\mathbf{x}) + \lambda \mathrm{L_k}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}_i\|^2 + \lambda(\mathbf{w}_k^T\mathbf{x} + b_k)$, where λ is the Lagrange multiplier.

By the first order necessary condition, $\nabla_{\mathbf{x}} \mathrm{L}(\mathbf{x}, \lambda) = (\mathbf{x} - \mathbf{x}_i) + \lambda \mathbf{w}_k = \mathbf{0}$ $\nabla_\lambda \mathrm{L}(\mathbf{x}, \lambda) = \mathbf{w}_k^T\mathbf{x} + b_k = 0$. We obtain $\mathbf{x} = \mathbf{x}_i - \frac{\mathbf{w}_k^T\mathbf{x}_i + b_k}{\|\mathbf{w}_k\|^2}\mathbf{w}_k$. Set $\|\mathbf{w}_k\|^2 = 1$, then $\|\mathbf{x} - \mathbf{x}_i\| = |\mathbf{w}_k^T\mathbf{x}_i + b_k|$. So, for kth line, the output of the distance layer is

$$d_{ik} = \mathrm{D}_k(\mathbf{x}_i) = |\mathbf{w}_k^T\mathbf{x}_i + b_k| = |\mathrm{L}_k(\mathbf{x}_i)| \, . \tag{1}$$

Distance from Point to Hyperbola. In a one-shot seismogram, the reflection wave pattern is a hyperbola. The equation is $-\left(\frac{x - x_{0,k}}{a_k}\right)^2 + \left(\frac{t - t_{0,k}}{b_k}\right)^2 - 1 = 0$ and

$$\mathrm{H}_k(\mathbf{x}) = b_k\sqrt{\left(\frac{x - x_{0,k}}{a_k}\right)^2 + 1} - (t - t_{0,k}) = 0 \; . \tag{2}$$

For reflection wave in the x-t space, the hyperbola is on the positive t and positive x space, so b_k is positive on (2).

The true distance from point to hyperbola is complicated. Here, we consider distance in time from the point (x_i, t_i) to the hyperbola $\mathrm{H}_k(\mathbf{x}) = 0$ as

$$d_{ik} = \left| b_k\sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1} - (t_i - t_{0,k}) \right| = \left|\mathrm{H}_k(\mathbf{x}_i)\right| \; . \tag{3}$$

3.2 Activation Function Layer

We use Gaussian basis function, and the activation function is defined as $Er_{ik} = f(d_{ik}) = 1 - \exp\left(-\frac{d_{ik}^2}{\sigma^2}\right)$. The Gaussian basis function controls the effect range of the point. Initially, we choose a larger σ, and σ is decreased as $\sigma/\log_2(1+iteration)$ by iterations. This choice shows that the range of effect is decreased when the iteration number is increased.

3.3 Total Error Layer

For each point $\mathbf{x}_\mathrm{i}$ the error function is defined as $Ec_i = C(\mathbf{Er}_i) = C(Er_{i1}, \cdots, Er_{ik}, \cdots, Er_{im}) = \prod_{1 \le j \le m} Er_{ij}$. Total error is zero when the distance between input $\mathbf{x}_\mathbf{i}$ and any object is zero, i.e., Er_{ik}=0, and Ec_i=0.

4 Parametric Learning Rules

In order to minimize total error Ec_i, we use gradient descent method to adjust parameters.

The parameters of line or hyperbola can be written as a parameter vector $\mathbf{p}_k$. $\mathbf{p}_k(\mathrm{t}+1) = \mathbf{p}_k(\mathrm{t}) + \Delta\mathbf{p}_k(\mathrm{t})$ (k = 1, 2, …, m) and

$$\Delta\mathbf{p}_k = -\beta\frac{\partial Ec_i}{\partial \mathbf{p}_k}, \tag{4}$$

where β is the learning rate. From (4) and by chain rule, $\triangle\mathbf{p}_k$ can be written as

$$\begin{aligned}\Delta\mathbf{p}_k &= -\beta\left(\frac{\partial Ec_i}{\partial d_{ik}}\right)\left(\frac{\partial d_{ik}}{\partial \mathbf{p}_k}\right) = -\beta\left(\frac{\partial Ec_i}{\partial Er_{ik}}\right)\left(\frac{\partial Er_{ik}}{\partial d_{ik}}\right)\left(\frac{\partial d_{ik}}{\partial \mathbf{p}_k}\right) \\ &= -\beta\left(\frac{Ec_i}{Er_{ik}}\right)\left(\frac{2d_{ik}}{\sigma^2}\right)(1 - f(d_{ik}))\left(\frac{\partial d_{ik}}{\partial \mathbf{p}_k}\right) \; .\end{aligned} \tag{5}$$

We derive $\frac{\partial d_{ik}}{\partial \mathbf{p}_k}$ for line and hyperbola as follows.

4.1 Learning Rule for Line

For a line, the parameter vector is $\mathrm{p}_k = [\ w_{k,1}\ \ w_{k,2}\ \ b_k\ ]^T = [\ \mathrm{w}_k^T\ \ b_k\ ]^T$ Thus, $\frac{\partial d_{ik}}{\partial \mathbf{p}_k} = \left[\left(\frac{\partial d_{ik}}{\partial \mathbf{w}_k}\right)^T \quad \frac{\partial d_{ik}}{\partial b_k}\right]^T = \left[\frac{\partial d_{ik}}{\partial w_{k,1}} \quad \frac{\partial d_{ik}}{\partial w_{k,2}} \quad \frac{\partial d_{ik}}{\partial b_k}\right]^T$. From (1), we can get

$$\frac{\partial d_{ik}}{\partial \mathbf{w}_k} = sign(d_{ik})\,\mathbf{x}_i \tag{6}$$

$$\frac{\partial d_{ik}}{\partial b_k} = sign(d_{ik}) \tag{7}$$

where $sign(d_{ik}) = \begin{cases} 1, & d_{ik} > 0 \\ 0, & d_{ik} = 0 \\ -1, & d_{ik} < 0 \end{cases}$. Hence, from (5), (6), and (7),

$$\Delta \mathbf{p}_k = [\Delta \mathbf{w}_k^T \quad \Delta b_k]^T = -\beta \left(\frac{Ec_i}{Er_{ik}}\right)\left(\frac{2d_{ik}}{\sigma^2}\right)(1 - f(d_{ik}))sign(d_{ik})[\mathbf{x}_i^{\ T}\ 1]^T . \tag{8}$$

Note that, in (8), $\Delta \mathbf{w}_k^T$ is proportional to $\mathbf{x}_i^{\ T}$, while Δb_k is not. That is, $\Delta \mathbf{w}_k^T$ is drastically affected by input scalar, but Δb_k is not. In order to solve this problem, we normalize the input data $[x \quad t]^T$ to satisfy

$$\mathrm{E}[\mathbf{x}] = \mathrm{E}[\mathbf{t}] = 0 \ \ and \ \ \mathrm{var}(\mathbf{x}) = \mathrm{var}(\mathbf{t}) = 1 \ . \tag{9}$$

After convergence, we obtain the parameter vector of normalized data, then we recover it to get parameter vector of original data. Without this normalization it is difficult to get the learning convergence.

4.2 Learning Rule for Hyperbola

For reflection wave, the parameter vector of hyperbola is $\mathbf{p}_k = [\ a_k\ \ b_k\ \ x_{0,k}\ \ t_{0,k}\ ]^T$. Thus, $\frac{\partial d_{ik}}{\partial \mathbf{p}_k} = \left[\frac{\partial d_{ik}}{\partial a_k} \quad \frac{\partial d_{ik}}{\partial b_k} \quad \frac{\partial d_{ik}}{\partial x_{0,k}} \quad \frac{\partial d_{ik}}{\partial t_{0,k}}\right]^T$. From (3),

$$\frac{\partial d_{ik}}{\partial a_k} = sign(d_{ik})\left(-\frac{b_k}{a_k}\right)\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 \Bigg/ \sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1}$$

$$\frac{\partial d_{ik}}{\partial b_k} = sign(d_{ik})\sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1} \tag{10}$$

$$\frac{\partial d_{ik}}{\partial x_{0,k}} = sign(d_{ik})\left(-\frac{b_k}{a_k}\right)\left(\frac{x_i - x_{0,k}}{a_k}\right) \Bigg/ \sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1}$$

$$\frac{\partial d_{ik}}{\partial t_{0,k}} = sign(d_{ik})$$

Then, from (5), and (10), we have

$$\Delta \mathbf{p}_k = [\Delta a_k \quad \Delta b_k \quad \Delta x_{0,k} \quad \Delta t_{0,k}]^T = -\beta\left(\frac{2}{\sigma^2}\right)\left(\frac{Ec_i d_{ik}}{Er_{ik}}\right)(1 - f(d_{ik}))sign(d_{ik}) \cdot \begin{bmatrix} \left(-\frac{b_k}{a_k}\right)\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 \Bigg/ \sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1} \\ \sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1} \\ \left(-\frac{b_k}{a_k}\right)\left(\frac{x_i - x_{0,k}}{a_k}\right) \Bigg/ \sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1} \\ 1 \end{bmatrix} \tag{11}$$

Also note here, input data scalar affects Δa_k, Δb_k and $\Delta x_{0,k}$. So data normalization by (9) and renormalization are also necessary for the hyperbola.

For seismic reflection wave pattern, in the geologic flat layer, we have $x_{0,k} = 0$ in (2). So the parameter vector $\mathbf{p}_k = [\ a_k \ \ b_k \ \ t_{0,k}\ ]^T$ and by (11) which implies parameter adjustment

$$\Delta \mathbf{p}_k = [\Delta a_k \quad \Delta b_k \quad \Delta t_{0,k}]^T = -\beta\left(\frac{2}{\sigma^2}\right)\left(\frac{Ec_i d_{ik}}{Er_{ik}}\right)(1 - f(d_{ik}))sign(d_{ik}) \cdot \begin{bmatrix} \left(-\frac{b_k}{a_k}\right)\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 \Bigg/ \sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1} \\ \sqrt{\left(\frac{x_i - x_{0,k}}{a_k}\right)^2 + 1} \\ 1 \end{bmatrix} \tag{12}$$

Similar to [6], in the learning process, we use two stage learning and the convergence can be fast. In the first stage, we only change the bias b_k of the line in (8), and ($x_{0,k}$, $t_{0,k}$) of the hyperbola in (11) or $t_{0,k}$ of the hyperbola in (12) until there is no significant change in the output error. In the second stage, we adjust all parameters of line in (8) and hyperbola in (11) or (12).

The flowchart of the learning system is shown in Fig. 5. The object number m is 2, one is line and the other is hyperbola. Initially set up random parameter vectors. Then input data and adjust the parameter vector as (8) and (11) or (12). Finally, if the average error is less than a threshold, E_{th}, then the learning stops.

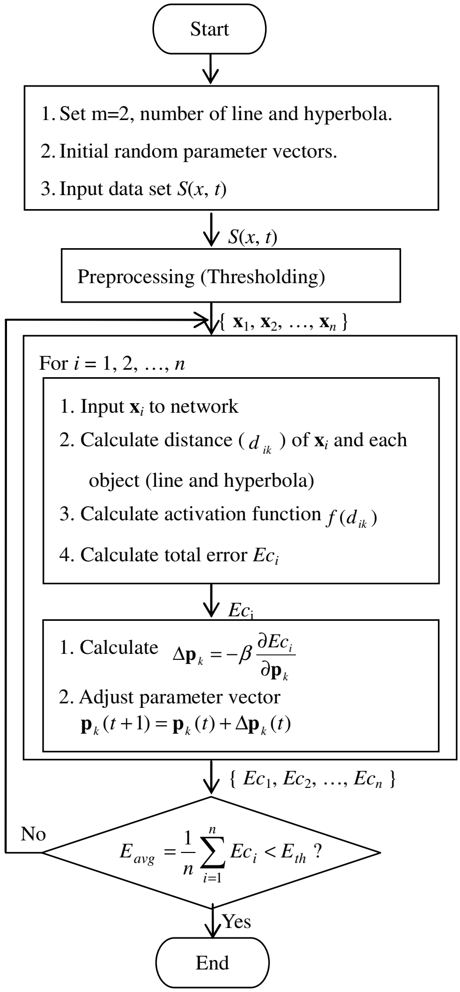

Fig. 5. Flowchart of the learning system

5 Seismic Experiments

The HTNN is applied to the simulated and real seismic data. In a simulated one-shot seismogram, the reflection layer is flat, that means $x_0 = 0$, so three parameters are detected for hyperbola. And in real seismic data, we have no prior geological knowledge, so four parameters are detected for hyperbola. In the experiments, the input data are in the image space and the results are shown in the *x-t* space.

5.1 Experiment on a Simulated One-Shot Seismogram

The image space of simulated one-shot seismogram in Fig. 1 is 64×512. After preprocessing, the input data in Fig. 6 have 252 points. Table 1 shows the detected parameters of line and hyperbola in the image space. The experimental results are shown in Fig. 6-7. We choose that β equals to 0.05, σ equals to $15/\log_2(1+iteration)$, and error threshold (E_{th}) equals to 10^{-5}. Fig. 6 shows the result of detection of direct wave and reflection wave in the *x-t* space. Fig. 7 shows the error versus iteration number, where the dotted line means it takes 12 iterations to change to stage two. Comparing the detection results with the original seismogram, the result of experiment is quite successful.

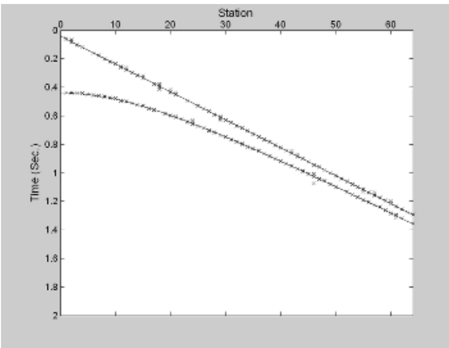

Fig. 6. Detection result: direct wave and reflection wave

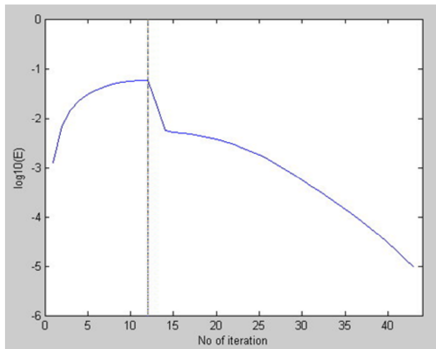

Fig. 7. Error versus iteration

5.2 Experiment on Real Seismic Data

Fig. 8 is the seismic data at Offshore Trinadad with 48 traces and 2050 points in each trace. The sampling rate is 0.004 seconds. The data are from Seismic Unix System developed by Colorado School of Mine [8]. After preprocessing, the input data in Fig. 9 have 755 points. Table 2 shows the detected parameters of line and hyperbola in the

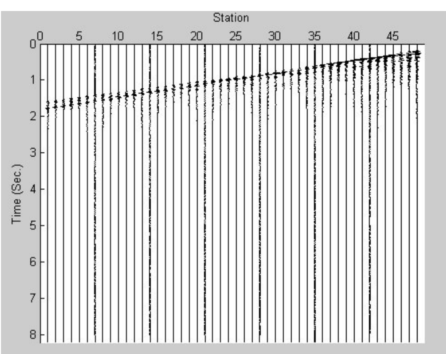

Fig. 8. Real seismic data at Offshore Trinadad

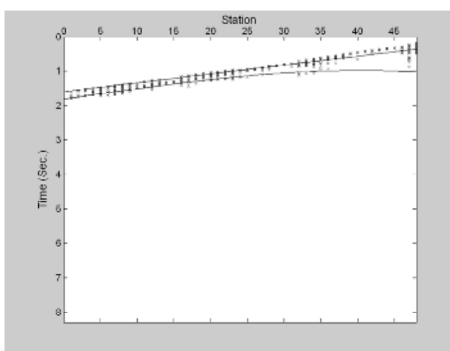

Fig. 9. Detection result: direct wave and reflection wave

image space. The results are shown in Fig. 8-10. We choose that β equals to 0.1, σ equals to $12/\log_2(1+iteration)$, and error threshold (E_{th}) equals to 2.5×10^{-4}. Fig. 9 shows the result of detection of direct wave and reflection wave in the *x-t* space. Fig. 10 shows the error versus iteration number, where the iteration number from stage one to stage two is 18.

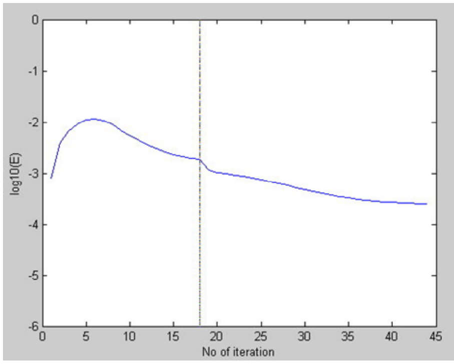

Fig. 10. Error versus iteration

Table 1. Parameters of line and hyperbola in Fig. 6 in the image space, 64×512

	w_1	w_2	b
Line	-0.040031	0.0079809	-0.082842
	a	b	t_0
Hyperbola	-21.176	-10.383	4.5614

Table 2. Parameters of line and hyperbola in Fig. 9 in the image space, 48×2050

	w_1	w_2		b
Line	0.04443	0.0068985		-2.7525
	a	b	x_0	t_0
Hyperbola	-28.371	-16.823	40.361	-37.19

6 Conclusions

HTNN is adopted to detect line pattern of direct wave and hyperbola pattern of reflection wave in a seismogram. The parameter learning rule is derived by gradient descent method to minimize the error. We use the direct line equation in the distance calculation from point to line. Also we define the vertical time difference as the distance from point to hyperbola that makes the learning feasible. In experiments, we get fast convergence in simulated data because three parameters are considered in the hyperbola detection. In real data, four parameters are in the hyperbola detection. There is no prior geological information, the detection result in line is good, but not in hyperbola. There may be 3 reasons: (1) input points are not many enough, (2) two objects are too close and affect each other, (3) there are reflections of deeper layers and affect the detection of the first layer reflection. Surely the detection results can provide a reference and improve seismic interpretation.

The result of preprocessing is quite critical for the input-output relation. More wavelet, envelope, and deconvolution processing may be needed in the preprocessing to improve the detection result.

Acknowledgments

This work was supported in part by the National Science Council, Taiwan, under NSC-94-2213-E-009-133. The authors would like to thank the use of Seismic Unix System developed by Colorado School of Mine.

References

1. Hough, P.V.C.: Method and Means for Recognizing Complex Patterns. U.S. Patent 3069654. (1962)
2. Duda, R.O., Hart, P.E.: Use of the Hough Transform to Detect Lines and Curves in Pictures. Comm. Assoc. Comput. Mach., Vol. 15. (1972) 11-15
3. Leavers, V.F.: Survey: Which Hough Transform. Computer Vision, Graphics, and Image Processing, Vol. 58, No. 2. (1993) 250-264
4. Huang, K.Y., Fu, K.S., Sheen, T.H., Cheng, S.W.: Image Processing of Seismograms: (A) Hough Transformation for the Detection of Seismic Patterns. (B) Thinning Processing in the Seismogram. Pattern Recognition, Vol. 18, No.6. (1985) 429-440
5. Dempsey, G.L., McVey, E.S.: A Hough Transform System Based on Neural Networks. IEEE Trans. Ind. Electron, Vol. 39. (1992) 522-528
6. Basak, J., Das, A.: Hough Transform Networks: Learning Conoidal Structures in a Connectionist Framework. IEEE Trans. on Neural Networks, Vol. 13, No. 2. (2002) 381-392
7. Basak, J., Das, A.: Hough Transform Network: A Class of Networks for Identifying Parametric structures. Neurocomputing, Vol. 51. (2003) 125-145
8. Yilmaz, O.: Seismic Data Processing. The Society of Exploration Geophysicists, Tulsa, (1987)